

Test Both The Models

Now that we have trained both the models' let's test both the models by loading the saved models. let's create another notebook for testing

Test The Model

The model is to be tested with different images to know if it is working correctly.

Import the packages and load the saved model

Import the required libraries

```
from keras.preprocessing import image
from tensorflow.keras.preprocessing.image import img_to_array
from tensorflow.keras.models import load_model
import numpy as np
```

Initially, we will be loading the fruit model. You can test it with the vegetable model in a similar way.

```
model = load_model("fruit.h5")
```

Load the test image, pre-process it and predict

Pre-processing the image includes converting the image to array and resizing according to the model. Give the pre-processed image to the model to know to which class your model belongs to.

```
x = image.img_to_array(img)
x = np.expand_dims(x,axis = 0)
```

```
pred = model.predict_classes(x)
```

```
pred
```

```
[1]
```

```
img = image.load_img('apple_healthy.JPG',target_size = (128,128))
```

The predicted class is 1

Application Building

After the model is built, we will be integrating it into a web application so that normal users can also use it. The new users need to initially register in the portal. After registration users can log in to browse the images to detect the disease.

In this section, you have to build

- HTML pages - front end
- Python script - Server-side script

Let's create a Python script using spyder IDE

Build Python Code

After the model is built, we will be integrating it into a web application so that normal users can also use it. The user needs to browse the images to detect the disease.

Activity 1: Build a flask application

Step 1: Load the required packages

```
import requests
from tensorflow.keras.preprocessing import image
from tensorflow.keras.models import load_model
import numpy as np
import pandas as pd
import tensorflow as tf
from flask import Flask, request, render_template, redirect, url_for
import os
from werkzeug.utils import secure_filename
from tensorflow.python.keras.backend import import_set_session
```

Step 2: Initialize the flask app and load the model

An instance of Flask is created and the model is loaded using load_model from Keras.

```
app = Flask(__name__)
|
#load both the vegetable and fruit models
model = load_model("vegetable.h5")
model1=load_model("fruit.h5")
```

Step 3: Configure the home page

```
#home page
@app.route('/')
def home():
    return render_template('home.html')
```

Step 4: Pre-process the frame and run

Pre-process the captured frame and give it to the model for prediction. Based on the prediction the output text is generated and sent to the HTML to display. We will be loading the precautions for fruits and vegetables excel file to get the precautions based on the output and return it to the HTML Page.

```

#prediction page
@app.route('/prediction')
def prediction():
    return render_template('predict.html')

@app.route('/predict',methods=['POST'])
def predict():
    if request.method == 'POST':
        # Get the file from post request
        f = request.files['image']
        # Save the file to ./uploads
        basepath = os.path.dirname(__file__)
        file_path = os.path.join(
            basepath, 'uploads', secure_filename(f.filename))
        f.save(file_path)
        img = image.load_img(file_path, target_size=(128, 128))
        x = image.img_to_array(img)
        x = np.expand_dims(x, axis=0)
        plant=request.form['plant']
        print(plant)
        if(plant=="vegetable"):
            preds = model.predict_classes(x)
            print(preds)
            df=pd.read_excel('precautions - veg.xlsx')
            print(df.iloc[preds[0]]['caution'])
        else:
            preds = model1.predict_classes(x)

            df=pd.read_excel('precautions - fruits.xlsx')
            print(df.iloc[preds[0]]['caution'])
    return df.iloc[preds[0]]['caution']

```

Run the flask application using the run method. By default, the flask runs on 5000 port. If the port is to be changed, an argument can be passed and the port can be modified.

```

if __name__ == "__main__":
    app.run(debug=False)

```

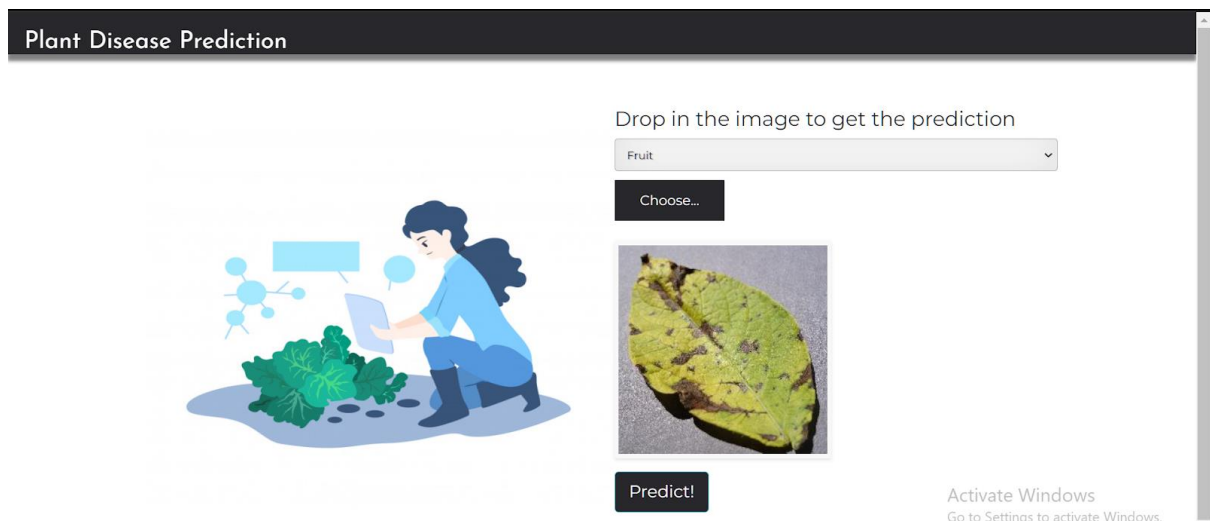
Build HTML Pages

Build the UI where a home page will have details about the application, a prediction page where a user is allowed to browse an image and get the predictions.

The home page looks like this.



After clicking on predict button, you will be redirected to the prediction page where you can browse the images.



Run The Code

Step 1: Run the application

In anaconda prompt, navigate to the folder in which the flask app is present. When the python file is executed the localhost is activated on 5000 port and can be accessed through it.

```
(base) F:\Projects\Plant Disease Detection\main>python app.py
```

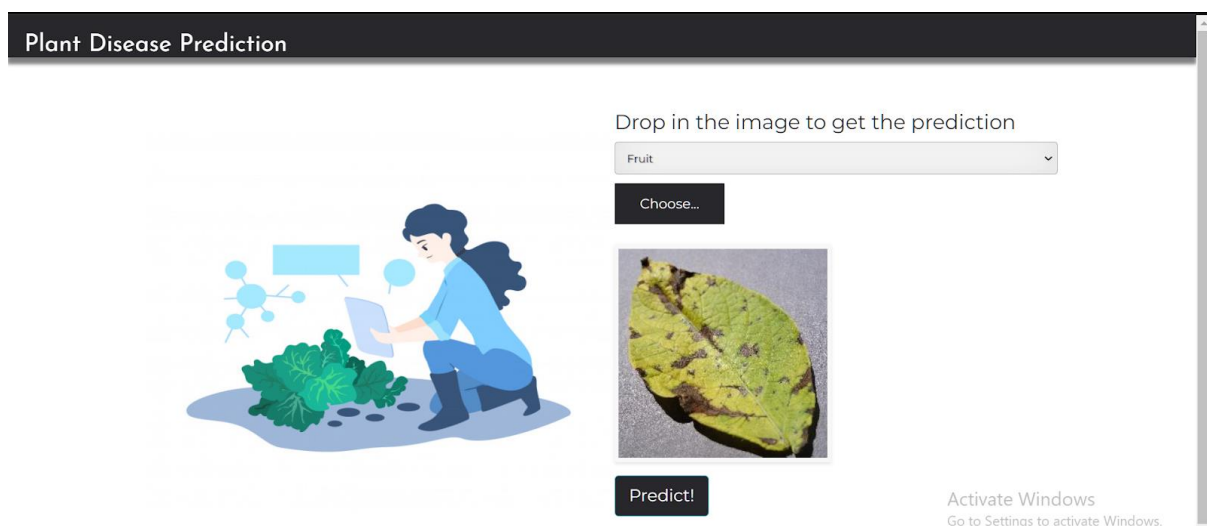
```
* Debugger is active!  
* Debugger PIN: 257-358-499  
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

Step 2: **Open the browser and navigate to localhost:5000 to check your application**

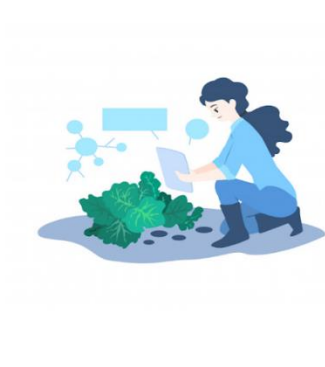
The home page looks like this.



After clicking on predict button, you will be redirected to the prediction page where you can browse the images.




Plant Disease Prediction



Drop in the image to get the prediction

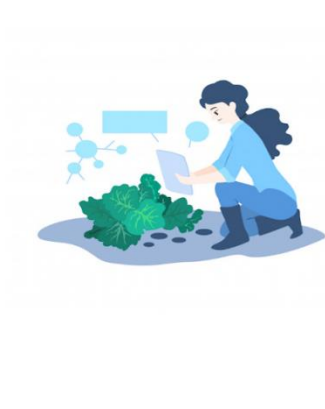
Vegetable

Choose...



Prediction: Oops!! Your tomato plant is infected by Septoria leaf spot. Removing the infected leaves immediately will curb the spread of infection. Organic and chemical fungicides with chlorothalonil are effective in treatment.


Plant Disease Prediction



Drop in the image to get the prediction

Fruit

Choose...



Prediction: Oops!! Your apple plant is infected by Black Rots. This infection is a fungal infection. To control black rot, remove the cankers by pruning at least 15 inches below the end and burn or bury them. Treating the sites with the antibiotic streptomycin or a copper-based fungicide will be helpful.

Train The Model On IBM

In this milestone, you will learn how to build Deep Learning Model Using the IBM cloud.

Register For IBM Cloud

To complete this project you must have an IBM account

IBM Account:

- Please click [here](#) to register for IBM
- Please click [here](#) to log in to IBM Account.

Watch the below video to register and login into your IBM account

<https://youtu.be/x6i43M7BAqE>

Train Model On IBM

Please watch the below video to train the model on IBM and integrate it with the flask Application.

<https://youtu.be/BzouqMGJ41k>

Ideation Phase

In this milestone you are expected to get started with the Ideation process.

Literature Survey On The Selected Project & Information Gathering

In this activity you are expected to gather/collect the relevant information on project usecase, refer the existing solutions, technical papers, research publications etc.

Prepare Empathy Map

In this activity you are expected to prepare the empathy map canvas to capture the user Pains & Gains, Prepare list of problem statements

Ideation

In this activity you are expected to list the ideas (at least 4 per each team member) by organizing the brainstorming session and prioritize the top 3 ideas based on the feasibility & importance.